Morphing on Clustering-Based Hierarchical Level-of-Detail with Bounded Error

Chih-Hao Hsu, Kuo-Chou Tseng, and Chin-Ho Cheng

Department of Computer Science and Information Engineering
Fu Jen Catholic University, Hsinchchuang, Taipei 24205, Taiwan, R. O. C.
E-mail: {konica86, alf85, chcheng}@csie.fju.edu.tw

Abstract

In this paper, the goal of morphing is to find intermediate models between two models corresponding to two consecutive levels of a level-of-detail (LOD) structure. By using the data structure of this LOD structure, we can easily find the correspondence between the source model and the destination model, and have a method to solve this 3D morphing problem. This method is tested, and our experimental results show that it is fast and efficient. In addition, the intermediate models generated by our morphing algorithm is also simplified.

Keywords: computer graphics, level of detail, morphing, simplified model, mesh

1. Introduction

In this paper, the purpose of morphing is to find the intermediate models between two models corresponding to two consecutive levels of a level-of-detail (LOD) structure, respectively. This LOD structure is constructed from the clustering-based algorithm shown in [12]. This LOD algorithm repeatedly executes the following three steps: clustering, boundary straightening, and triangulation. Each iteration corresponds to a pass and generates a one-level object. Initially, the first level object is the original model of input data. The first iteration of this LOD algorithm performs on the original model, and we can get a simplified model with smaller boundary information. Then, this simplified model is used as the input of the second iteration of this LOD algorithm, and another simplified model with smaller boundary information is generated. In general, any iteration takes as input data the simplified model generated by its previous one iteration, and generates a new simplified model with smaller boundary information, which will be used as the input data of its next iteration. Then, we can have a series of model data, and each level object preserves the feature of the original input model. Objects in this series of models are from the most detailed object to the least detailed object. Because of the behavior of this LOD algorithm, showing two consecutive models (corresponding to two consecutive levels in the LOD tree structure) has popping conditions. That is, the transition between two consecutive models is not enough smooth. Hence, in this paper, we propose a method to insert intermediate models into any two consecutive models so as to make smooth transition between these two consecutive models.

This paper is organized as follows. In Section 2, we describe the related work. In Section 3, our morphing method is described. In Section 4, mesh morphing is discussed. Experimental results are illustrated in Section 5. Finally, conclusions are given in Section 6.

2. Related Work

Lazarus and Verroust [6] give an excellent survey of previous work on the 3D morphing problem. Most methods for morphing 3D objects use either discrete or combinatoric representations for the objects themselves. Discrete representations typically voxelize objects or their distance functions and aim to extend 2D morphing algorithms to 3D. Lerious et at. [8] used fields of influence of 3D primitives to warp-volumes. Hughes [3] proposed a method working in the Fourier domain. Payne [11] described a distance-field volumetric cross-dissolving technique. The alternative is to work directly on boundary representations such as polygonal meshes or patch complexes [5, 9, 10, 1, 4, 2, 7].

3. Our Morphing Method

The morphing method presented in this paper is to find intermediate models between two consecutive level models in an LOD tree structure. The LOD tree structure is constructed from the Tseng and Cheng's clustering-based LOD algorithm [12]. In this LOD algorithm, it repeatedly produces simplified models. Each iteration of this algorithm performs three steps: clustering, border straightening, and triangulation. For each iteration, our morphing method works on two models, which are the input and output models of the boundary straightening process, respectively. Suppose that the input model is *M* and that the output model is *M'*. Our morphing method is based on the following properties:

(1) The number of meshes in *M* is equal to that of meshes in *M'*.
(2) For any mesh *m* in *M*, there is a corresponding mesh *m'* in *M'*, which is the result of the boundary straightening process.

Therefore, the kernel of our morphing method in this paper really performs mesh morphing between *m* and *m'*. That is, it is to find intermediate meshes from *m* to *m'*. Because of the behavior of our morphing algorithm, all intermediate meshes between any mesh and its corresponding border-straightened mesh have the same number of vertices, i.e., they have the same topology. If we perform the triangulation process on these intermediate meshes, each intermediate mesh will be triangulated into the same number of triangles. In such a way, these intermediate meshes have no simplification effect. In order to have simplification effect on these intermediate meshes, we apply the border straightening process to each intermediate mesh, and take the output of the border straightening process as the input of the triangulation process. Finally, we have got a set of triangles for each intermediate mesh. In the next section, we will describe the mesh morphing.

4. Mesh Morphing

Before describing the meshing morphing method, we first discuss border transition in the following section. Then, we use this border transition technique to develop our mesh morphing method.

4.1 Border Transition

Suppose that we are given two borders B_s and B_e, where border B_s has the vertex sequence $b_1, b_2, \cdots, b_n$ with b_1 and b_n being the start and end vertices, respectively, and border B_e is the border connecting the start and end vertices of border B_s. (That is, border B_e is the line segment $\overline{b_1 b_n}$.) See Fig. 1. In this section, we will show how to find the intermediate transition from the border B_s to the border B_e.

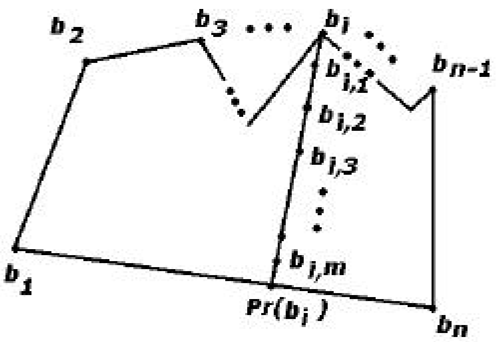

Fig. 1

First, we define the projection of vertex b_i, for $i=2, 3, \ldots, n-1$, onto the line segment connecting the start and end vertices of border B_s, the line segment $\overline{b_1 b_n}$. Let it be denoted by $\Pr(b_i)$. The projection point $\Pr(b_i)$ is a point on the line segment $\overline{b_1 b_n}$ which is nearest to vertex b_i. Then, according to the geometric properties, one of the following cases about the projection point $\Pr(b_i)$ should occur: (1) the line segment $\overline{b_i \Pr(b_i)}$ is perpendicular to the line segment $\overline{b_1 b_n}$; (2) the point $\Pr(b_i)$ is either b_1 or b_n. By simple computation, we can easily get the projection point $\Pr(b_i)$.

Then, for any line segment $\overline{b_i \Pr(b_i)}$, for $i=2, 3, \ldots, n-1$, we choose m partition points to partition the line segment $\overline{b_i \Pr(b_i)}$ into $m+1$ pieces, each of which has equal length. Let these m partition points be $b_{i,1}$, $b_{i,2}$, $\ldots b_{i,m}$ which are ordered such that point $b_{i,1}$ is nearest to b_1 and point $b_{i,m}$ is nearest to $\Pr(b_i)$. (See Fig. 1.) Hence, we can get m new borders, say $B_1, B_2, \ldots, B_m$, where the border B_j has the vertex sequence $b_1, b_{1,j}, b_{2,j}, \ldots, b_{n-1,j}, b_n$, for $j=1,2,\ldots,m$. Border B_j also has vertices b_1 and b_n to be its start and end vertices, respectively. We call these m borders $B_1, B_2, \ldots B_m$ the m intermediate borders transition from the border B_s to the border B_e. These m borders are what we want. When m is large enough, showing borders $B_s, B_1, B_2, \ldots B_m, B_e$ has smooth transition from the border B_s to the border B_e.

4.2 Our Mesh Morphing Method

Now, our mesh morphing method is presented. Suppose that the mesh before the boundary straightening process woks on is *M* and that the mesh after the boundary straightening process works on is *M'*. In this section, we will show how to find the smooth transition from the mesh *M* to the mesh *M'*. That is, we will find some intermediate meshes whose shapes change from *M* to *M'* smoothly. Assume that the mesh *M'* consists of *n* edges e_1, e_2, …, e_n, which are ordered in clockwise (or counterclockwise) sequence. For each edge of the border on M', it has a corresponding border piece on M which is straightened into this edge (on M'). Now, let us give some notations. For each edge e_i ($i= 1,2,\ldots,n$) of the mesh M', its corresponding border on M is denoted by E_i. We note that E_i may be the edge e_i. In general, the border E_i is part of the boundary of the mesh M and contains one or more consecutive boundary edges of the mesh M. Then the border sequence $E_1, E_2, \ldots E_n$ forms the boundary of the mesh M. By taking advantage of the relationship between border E_i and edge e_i, we can find intermediate meshes from mesh M to mesh M' as follows.

If the number of intermediate meshes from mesh M to mesh M' that we need is m, let these intermediate meshes from mesh M to mesh M' be $M_1, M_2, \ldots M_m$. In this paper, the boundary of any intermediate mesh is constructed from the m intermediate borders from E_i to e_i (described in Section 4.1), for $i=1,2,\ldots,n$. Assume that the m intermediate borders from E_i to e_i, for $i=1,2,\ldots,n$, are $B_{i,1}, B_{i,2}, \ldots, B_{i,m}$. The boundary of any intermediate mesh M_i, for $i=1,2,\ldots,m$, is formed by n intermediate borders $B_{1j}, B_{2j}, \ldots, B_{nj}$, which are in clockwise (or counterclockwise) sequence. Once we have the boundaries of all the m intermediate meshes from mesh M to mesh M', smooth transition effect from M to M' can be obtained.

5. Experimental Results

We use two test models for experiment. The platform is Sun UltraSparc 350MHz with one CPU installed. The experimental results are illustrated in Sections 5.1 and 5.2.

5.1 Experiment 1

The input model is a cow model composed of 5804 triangles (Fig. 2). In this experiment, we run our morphing algorithm to find 10 intermediate models from the original (cow) model to a simplified model, which is obtained from the result of the first iteration (or pass) of running the Tseng and Cheng's LOD algorithm [12]. We note that the tolerance rate used in this Tseng and Cheng's LOD algorithm is $\pi/5$. In this experiment, we call this simplified model the destination model. This experimental result is shown in Table 1.

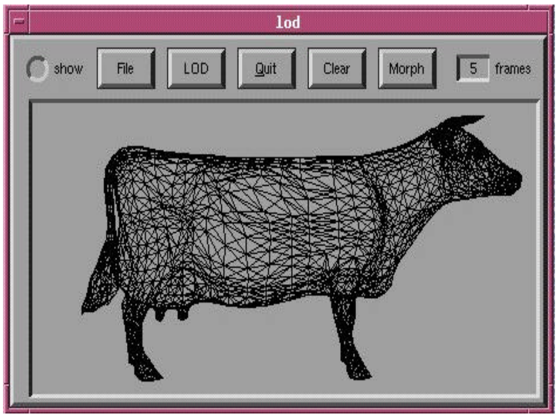

Fig.2.The original cow model with 5804 triangles.

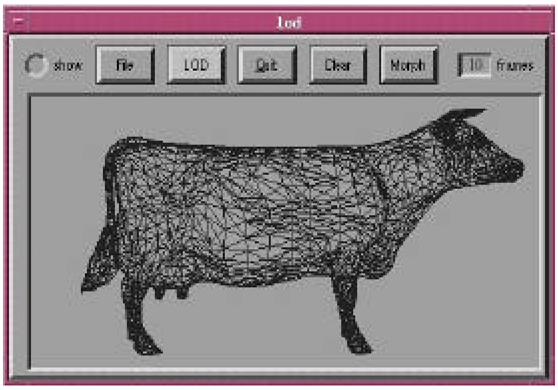

Fig. 3. The original cow model

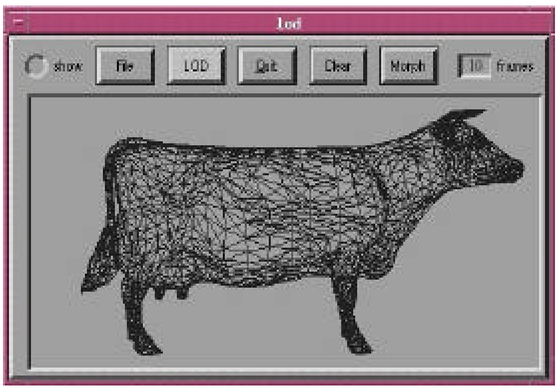

Fig. 4. Intermediate model 1

Model	No. of Triangles	Reduction Rate	Time
Original model	5804	0	0
Intermediate model 1	5664	97%	67
Intermediate model 2	5402	93%	70
Intermediate model 3	5248	90%	73
Intermediate model 4	5092	87%	77
Intermediate model 5	4872	83%	80
Intermediate model 6	4313	74%	83
Intermediate model 7	3880	67%	87
Intermediate model 8	3351	58%	89
Intermediate model 9	3024	52%	92
Intermediate model 10	2344	40%	96
Destination model	1750	30%	118

Table 1. (time unit: second)

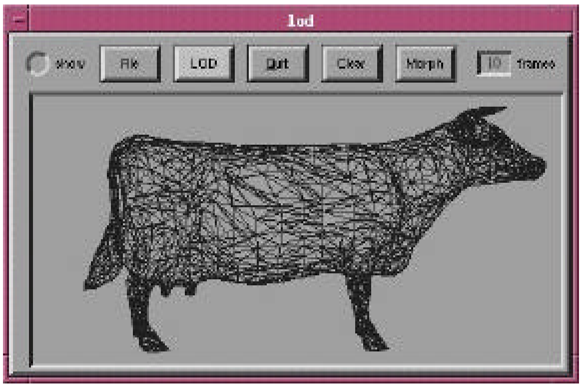

Fig.5. Intermediate model 2

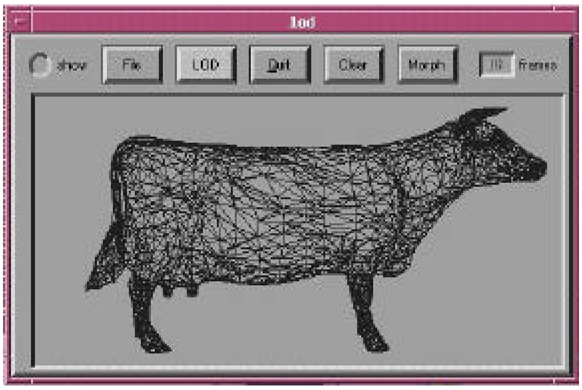

Fig. 6. Intermediate model 3

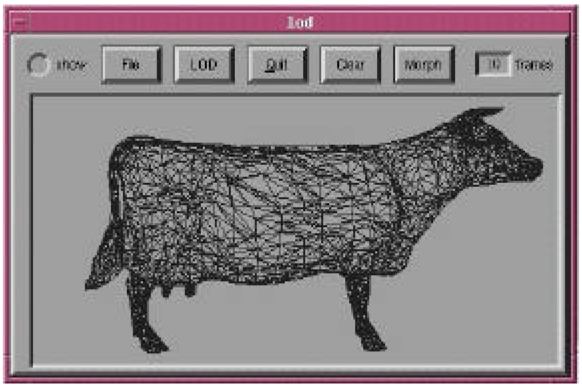

Fig. 7. Intermediate model 4

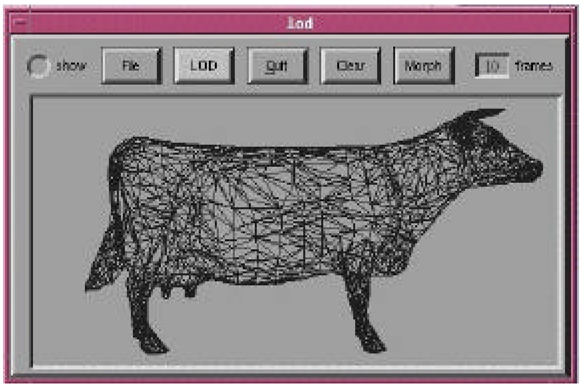

Fig. 8. Intermediate model 5

Table 1 shows that the Tseng and Cheng’s LOD algorithm [12] is run at time 0, and at time 118, the destination model is generated. At time 67, the intermediate model 1 is generated, and, after 3 seconds, the intermediate model 2 is generated at time 70. The last intermediate model , the intermediate model 10, is generated at time 96. It indicates that the time required between two consecutive intermediate models generating is about 2 ~ 4 seconds. From Table1, we know that the reduction rates of the 10 intermediate models decrease from 97% to 40 %, where the reduction rate of our destination model is about 30%. Furthermore, it shows that our morphing algorithm is fast and efficient. In order to illustrate the shape changing from the original (cow) model to the destination model, we show the original model, 10 intermediate models, and destination model in Fig. 3 ~ Fig. 14

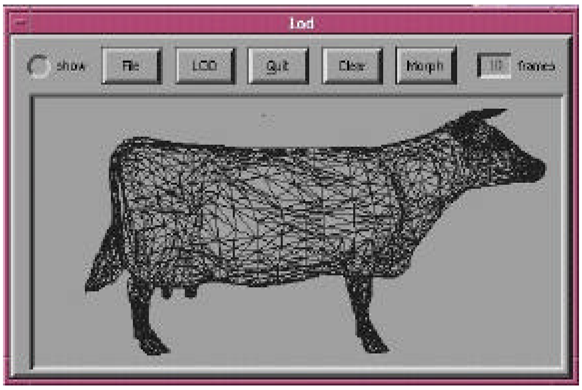

Fig. 9. Intermediate model 6

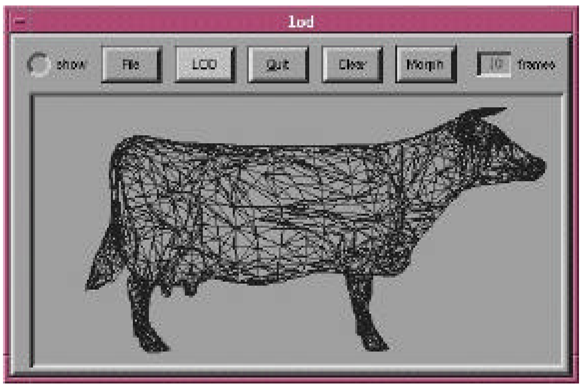

Fig. 10. Intermediate model 7

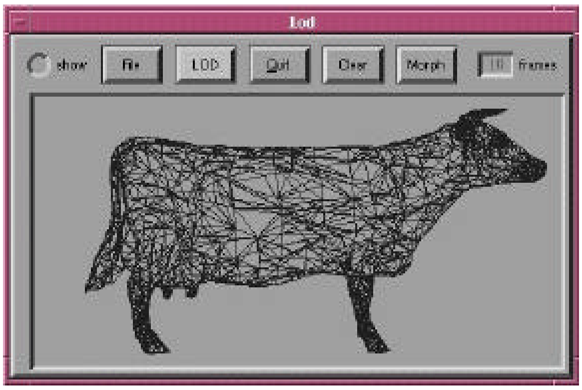

Fig. 11. Intermediate model 8

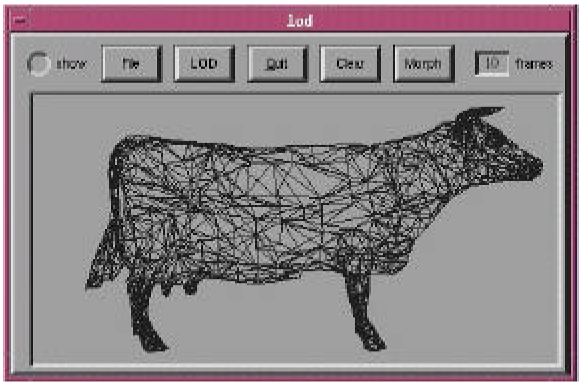

Fig. 12. Intermediate model 9

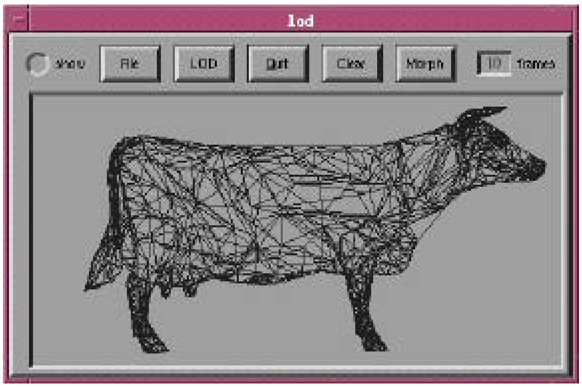

Fig. 13. Intermediate model 10

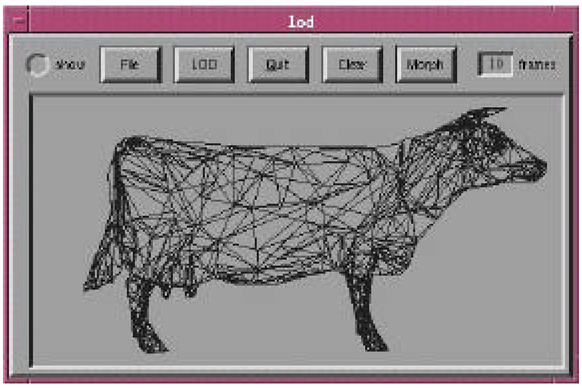

Fig. 14.The destination model

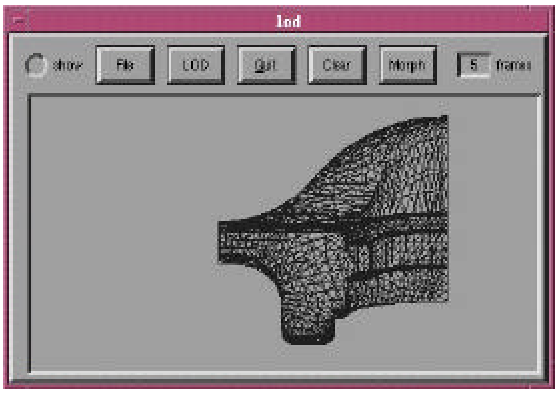

Fig. 15. The Fandisk model

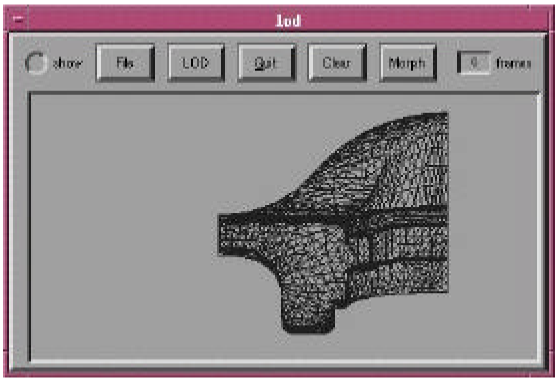

Fig. 16. The original Fandisk model

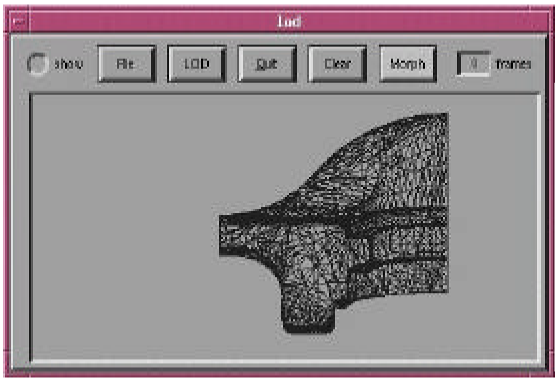

Fig. 17. Pass 1 Model

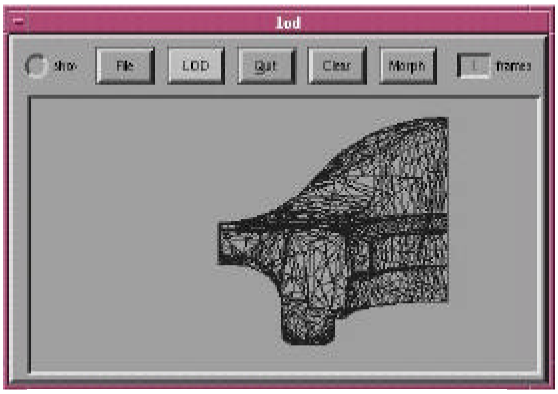

Fig. 18. The intermediate model between pass 1 model and pass 2 model

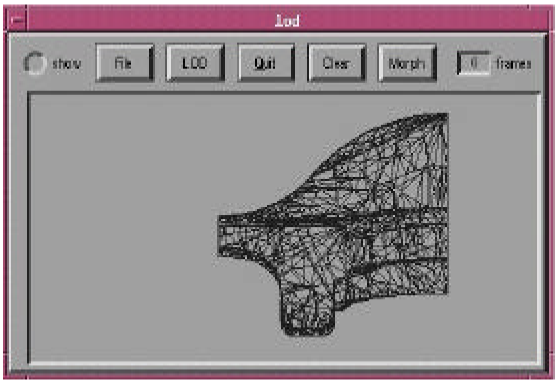

Fig. 19. Pass 2 model

Model	No. of Triangles	Reduction Rate	Time
Original Model	3392	0	0
Pass 1 model	3254	96%	67
Pass 2 model	1755	52%	93

Table 2. (time unit: second)

Model	No. of Triangles	Reduction Rate	Time
Original model	3392	0	0
Pass 1 model	3254	96%	67
Intermediate model	2369	70%	71
Pass 2 model	1755	52%	96

Table 3. (time unit: second)

5.2 Experiment 2

The second test model is called 'Fandisk' with 3392 triangles (Fig. 15). We run the Tseng and Cheng's LOD algorithm [12], and get two simplified models obtained from pass 1 and pass 2, respectively. (See Table 2.) In this experiment, we find one intermediate model between these two simplified models. The experimental result is illustrated in Table 3.

Table 3 shows that this Tseng and Cheng's LOD algorithm [12] is run at time 0, pass 1 model is generated at time 67, and the intermediate model is generated at time 71. In other words, it takes 4 seconds to generate the intermediate model after pass 1 model is generated. From Table 3, the reduction rates of pass 1 model and pass 2 model are 96% and 52%, respectively. The reduction rate of the intermediate model is 70% and lies between the reduction rates of pass 1 model and pass 2 model. It also shows that our morphing algorithm is fast and efficient. In order to show the shape changing form the original (Fandisk) model to the model generated by pass 2, we illustrate the original model, pass 1 model, this intermediate model, and pass 2 model in Fig. 16 ~ Fig. 19.

6. Conclusions

We have proposed a method to solve a 3D morphing problem on a LOD tree structure. This LOD structure is constructed from the Tseng and Cheng's LOD algorithm [12]. This algorithm repeatedly executes three steps: clustering, border straightening, and triangulation. Our morphing algorithm really performs mesh morphing. It is executed in the border straightening step of each iteration of this LOD algorithm. In the border straightening step, our morphing algorithm keeps all the meshes of the input model, and all their corresponding border-straightened meshes. Then, for each input mesh kept by our morphing algorithm, find the intermediate meshes between this input mesh and its corresponding border-straightened mesh. Because of the behavior of our morphing algorithm, all intermediate meshes between any mesh and its corresponding border-straightened mesh have the same number of vertices. That is, they have the same topology. If we perform the triangulation process on these intermediate meshes, each intermediate mesh will be triangulated into the same number of triangles. In such a way, these intermediate meshes have no simplification effect. In order to have simplification effect on these intermediate meshes, we apply the border straightening process to each intermediate mesh, and take the output of the border straightening process as the input of the triangulation process. Finally, we have got a set of triangles for each intermediate mesh. This effect can be seen from the reduction rates of the intermediate models in our experimental results. In other words, out morphing algorithm not only gets a set of intermediate models, but also has simplification effect. Our experimental result also shows that our morphing algorithm is fast and efficient.

References

[1] Eyal Carmel and Daniel Cohen-Or, Warp-guided object-space morphing, *The Visual Computer*, 13, 9-10 (1998), 46-478.

[2] Arthur Gregory, Andrei State, Ming C. Lin,Dinesh Manocha, Mark A. Livingston, Feature-based Surface Decomposition for Correspondence and Morphing between Polyhedra, Tech. Rep. TR98-014, Department of Computer Science, University of North Carolina at Chapel Hill, Apr. 14, 1998.

[3] Hughes, J. F., Scheduled Fourier volume morphing, Computer Graphics (*SIGGRAPH' 92 Proceedings*), 43-46, 1992.

[4] Kanai, T., Suzuki, H., and Kimura, F., Three-dimensional geometric metamorphosis based on harmonic maps, *The Visual Computer*,

14, 4 (1998), 144-176.

[5] J. Kent, W. Carlson, and R. Parent, Sharp Transformation for Polyhedral Object, *SIGGRAPH 92 Conference Proceedings*, pp. 47-54, 1992.

[6] Lazarus, F., and Verroust, A., Three-dimensional metamorphosis: a survey, *The Visual Computer*, 14 (1998), 373-389.

[7] Aaron W. F. Lee, David Dobkin, Wim Sweldens, Peter Schroder, Multiresolution Mesh Morphing, *Computer Graphics (SIGGRAPH'99)*, pp. 343-350, 1999.

[8] A. Lerios, C. Garfinkle, and M. Levoy, Feature-Based Volume Metamorphosis, *SIGGRAPH 95 Conferecce Proceedings*, pp.449 – 456, 1995.

[9] Sederberg, T. W., Gao, P., Wang, G., and Mu, H., 2D shape blending: an intrinsic solution to the vertex path problem, *Computer Graphics (SIGGRAPH'93 Proceedings)*, Vol.27, 15-18, Aug. 1993.

[10] Shapira, M., and Rappoport, A., Shape Blending Using the Star-Skeleton Representation, *IEEE Computer Graphics and Applications*, 15, 2 (1995), 44-50.

[11] Payne, B. A., Toga, A. W., Distance field manipulation of surface models, *IEEE Computer Graphics and Applications*, 12, 1 (Jan. 1992), 65-71.

[12] K. C. Tseng and C. H. Cheng, Clustering Based Hierarchical Level-of- Detail with Bounded Error, *Proceedings of 1998 International Computer Symposium, Workshop on Computer graphics and Virtual Reality,* pp. 97-103, Dec. 1998, N.C.K.U., Tainan, Taiwan, R.O.C.